Appears in *Proceedings of the 38th Annual ORSNZ Conference*

Lagrangian Relaxation and Enumeration for Solving Constrained Shortest-Path Problems

W. Matthew Carlyle
R. Kevin Wood
Operations Research Department
Naval Postgraduate School
Monterey, California, USA

Abstract

Recently published research indicates that a vertex-labeling algorithm based on dynamic-programming concepts is the most efficient procedure available for solving constrained shortest-path problems (CSPPs), i.e., shortest-path problems with one or more side constraints on the total "weight" of the optimal path. However, we investigate an alternative procedure that Lagrangianises the side constraints, optimises the resulting Lagrangian function and then closes any duality gap through enumeration of near-shortest paths. These paths are measured with respect to Lagrangian-modified edge lengths, and "near-shortest" implies ϵ-optimal, with ϵ equal to the duality gap. Our recently developed procedure for enumerating near-shortest-paths leads to an algorithm for CSPP that, empirically, proves to be an order of magnitude faster than the most recent vertex-labeling algorithm.

1 Introduction

Shortest-path problems in networks with non-negative edge lengths (or with some negative-length edges, but no negative-length cycles) can be solved easily, in polynomial time. However, if each edge possesses a weight in addition to its length, and if a single side constraint is placed on the optimal path's total weight, the problem becomes NP-complete (Garey and Johnson [13], p. 214). Multiple edge weights and weight limits may be defined, and we call the general problem the *constrained shortest-path problem* (CSPP). This paper describes and demonstrates an improved method for solving this problem.

CSPP is only NP-complete in the weak sense and admits a dynamic-programming solution procedure (Joksch [16]). However, dynamic programming (DP) can be unacceptably slow in practice, and vertex-labeling algorithms based on DP concepts have supplanted straightforward DP implementations (e.g., Aneja et al. [2], Dumitrescu and Boland [10]). Other potentially useful techniques include branch

and bound using a Lagrangian-based bound (Beasley and Christofides [4]), and Lagrangian relaxation coupled with K-shortest path enumeration (Handler and Zang [15]).

Dumitrescu and Boland [10] indicate that a vertex-labeling (label-setting) algorithm combined with several preprocessing techniques may be the most efficient technique currently available for CSPP. However, we have developed a new algorithm (Carlyle and Wood [7]) for enumerating near-shortest paths (NSPs), i.e., all paths that are within ϵ units of being shortest for a prespecified $\epsilon \geq 0$. This *NSP algorithm* can be used as a subroutine to solve the K-shortest-paths problem orders of magnitude faster than previous methods. Consequently, *Lagrangian relaxation plus enumeration* (LRE), similar to Handler and Zang [15], requires reevaluations as an alternative procedure for solving CSPP. That is the precise purpose of this paper.

We note that the paradigm of "near-shortest paths" is, in fact, more appropriate in the context of LRE for CSPP than is "K-shortest paths" as used in [15]. This issue is discussed in more detail later.

CSPP arises in a number of real-world contexts. The most well-known application may be column-generation for generalized set-partitioning models of crew-scheduling and crew-rostering problems, especially in the airline industry (e.g., Gamache et al. [12], Vance et al. [23]). Another important application is the minimum-risk routing of military aircraft and other military "vehicles" (e.g., Boerman [6], Latourell, et al. [19], Lee [20], Zabarankin et al. [25]). Additional applications include signal routing in communications networks having quality-of-service guarantees (see Korkmaz and Krunz [18] and the references therein), signal compression (Nygaard et al. [22]) and numerous transportation problems (e.g., Nachtigall [21], Kaufman and Smith [17]).

In the remainder of the paper, we first define CSPP precisely, and then describe the basic LRE solution approach. Next, we provide an overview of the NSP algorithm that makes our LRE approach viable, followed by a detailed, pseudo-code description of the enumeration portion of the LRE algorithm. Optimisation of the Lagrangian function is not discussed in detail because the relevant techniques are well known; a number of references are provided, however. Preliminary computational results are then presented for singly constrained shortest-path problems on large grid networks. These problems mimic the structure of some difficult problems solved by Dumitrescu and Boland [10], and therefore allow for useful comparisons of techniques. The paper concludes with a summary and directions for further research.

2 Basic Problem and Approach

We are given a directed network $G = (V, E)$, where V represents a set of vertices, and E represents a set of directed edges (u, v) connecting distinct vertices $u,\ v \in V$. Each edge $(u, v) \in E$ has a length $c_{uv} \geq 0$ and one or more weights, $f_{iuv} \geq 0$, for $i \in I$. (Non-negativity of lengths and weights is not an absolute requirement, but this assumption simplifies the following discussion.) Two distinct vertices $s, t \in V$ are defined, as well as a limit $g_i \geq 0$ on path weight for each $i \in I$. The *constrained*

shortest-path problem (CSPP) is to find a loopless, directed, s-t path p, which we denote here through its edge set $E_p \subset E$, such that $\sum_{(u,v)\in E_p} f_{iuv} \leq g_i$ for all $i \in I$ and such that $\sum_{(u,v)\in E_p} c_{uv}$ is minimised.

Let A denote the standard vertex-edge incidence matrix for G, and let $b_s = 1$, $b_t = -1$ and $b_v = 0$ for all $v \in V\backslash\{s,t\}$. Then, CSPP may be written as an integer program (Ahuja et al. [1], p. 599),

$$\begin{aligned}
\textbf{CSPIP} \quad z^* \;=\; & \min_{\mathbf{x}} \mathbf{cx} && (1)\\
& \text{s.t. } A\mathbf{x} \;=\; \mathbf{b} && (2)\\
& F\mathbf{x} \;\leq\; \mathbf{g} && (3)\\
& \mathbf{x} \;\geq\; \mathbf{0} \text{ and integer,} && (4)
\end{aligned}$$

where equations (3) are the side constraints, and where $x^*_{uv} = 1$ if edge (u,v) is in the optimal path, and $x^*_{uv} = 0$, otherwise, (The problem's structure leads to binary solutions without explicit constraints $\mathbf{x} \leq \mathbf{1}$.)

CSPIP would be easy to solve—it would just be a shortest-path problem—if not for the side constraints $F\mathbf{x} \leq \mathbf{g}$. We expect to have only a few such constraints, say one to ten, and it therefore seems reasonable to base a solution procedure on relaxing them. Using the standard theory of Lagrangian relaxation, we know that for any row vector $\boldsymbol{\lambda} \geq \mathbf{0}$, appropriately dimensioned,

$$\begin{aligned}
z^* \;\geq\; \underline{z}(\boldsymbol{\lambda}) \;=\; & \min_{\mathbf{x}} \mathbf{cx} \;+\; \boldsymbol{\lambda}(F\mathbf{x} - \mathbf{g}) && (5)\\
& \text{s.t. } A\mathbf{x} \;=\; \mathbf{b} && (6)\\
& \mathbf{x} \;\geq\; \mathbf{0} \text{ and integer.} && (7)
\end{aligned}$$

We can then rewrite the objective function, and optimise the Lagrangian lower bound $\underline{z}^*$ through

$$\begin{aligned}
\textbf{CSPLR} \quad \underline{z}^* \;=\; & \max_{\boldsymbol{\lambda}\geq\mathbf{0}} \underline{z}(\boldsymbol{\lambda}) && (8)\\
\;=\; & \max_{\boldsymbol{\lambda}\geq\mathbf{0}} \min_{\mathbf{x}} (\mathbf{c} + \boldsymbol{\lambda}F)\mathbf{x} - \boldsymbol{\lambda}\mathbf{g} && (9)\\
& \text{s.t. } A\mathbf{x} \;=\; \mathbf{b} && (10)\\
& \mathbf{x} \;\geq\; \mathbf{0} \text{ and integer.} && (11)
\end{aligned}$$

For any fixed $\boldsymbol{\lambda} \geq 0$, computing $\underline{z}(\boldsymbol{\lambda})$ simply requires the solution of a shortest-path problem with Lagrangian-modified edge lengths.

The outer maximisation over $\boldsymbol{\lambda}$ can be solved via bisection search for one side constraint (Fox and Landi [11]), coordinate search for a few side constraints (e.g., DeWolfe, Stevens and Wood [9]), subgradient optimisation (Beasley and Christofides [4] describe this for CSPP), or even through a linear-programming master problem as in Benders decomposition (Benders [5]). In this preliminary research, we only study singly constrained problems, so bisection search suffices to maximise $\underline{z}(\boldsymbol{\lambda})$ nearly exactly.

Often, in the process of optimising $\underline{z}(\boldsymbol{\lambda})$, one finds a solution $\hat{\mathbf{x}}$ that is feasible with respect to the relaxed constraints (3) as well as to constraints (2) and (4). Specifically, if F is a non-negative matrix, by making the vector $\boldsymbol{\lambda}$ sufficiently large, violation of the constraints $F\mathbf{x} \leq \mathbf{g}$ is discouraged, and a feasible solution

typically results. We will assume we have found a feasible solution $\hat{\mathbf{x}}$, and can therefore compute the upper bound $\bar{z} \equiv \mathbf{c}\hat{\mathbf{x}} \geq z^*$.

Now, given $\bar{z}$, and an apparently "good" vector $\boldsymbol{\lambda}$, the problem of identifying $\mathbf{x}^*$ maybe be viewed as one of simple enumeration:

Theorem 1 *Let $\hat{X}(\boldsymbol{\lambda}, \bar{z})$ denote the set of feasible solutions to* **CSPLR** *such that $\mathbf{c}\hat{\mathbf{x}} + \boldsymbol{\lambda}(F\hat{\mathbf{x}} - \mathbf{g}) \leq \bar{z}$. Then, $\mathbf{x}^* \in \hat{X}(\boldsymbol{\lambda}, \bar{z})$. That is, an optimal solution $\mathbf{x}^*$ to* **CSPIP** *can be identified by enumerating $\hat{X}(\boldsymbol{\lambda}, \bar{z})$ and selecting*

$$\mathbf{x}^* \in \underset{\mathbf{x} \in \hat{X}(\lambda, \bar{z}) | F\mathbf{x} \leq \mathbf{g}}{\operatorname{argmin}} \mathbf{cx} \tag{12}$$

Proof: Since $F\mathbf{x}^* \leq \mathbf{g}$ and $\boldsymbol{\lambda} \geq \mathbf{0}$, this follows from the facts that (i) $\mathbf{c}\mathbf{x}^* + \boldsymbol{\lambda}(F\mathbf{x}^* - \mathbf{g}) \leq z^*$, and (ii) $z^* \leq \bar{z}$. ∎

This theorem is valid for any $\boldsymbol{\lambda} \geq \mathbf{0}$, but it is easy to devise examples that show how an optimal or near-optimal $\boldsymbol{\lambda}$ for **CSPLR** can exponentially reduce the size of $\hat{X}(\boldsymbol{\lambda}, \bar{z})$ and thereby exponentially reduce computational workload.

Note that Theorem 1 implies that we may need to enumerate each path, represented here by $\hat{\mathbf{x}}$, such that $(\mathbf{c} + \boldsymbol{\lambda}F)\hat{\mathbf{x}} - \boldsymbol{\lambda}\mathbf{g} \leq \bar{z}$. That is, if $\mathbf{x}^*_{\boldsymbol{\lambda}}$ solves the shortest-path problem given the edge-length vector $\mathbf{c}+\boldsymbol{\lambda}F$, and $\underline{z}(\boldsymbol{\lambda}) \equiv (\mathbf{c}+\boldsymbol{\lambda}F)\hat{\mathbf{x}}^*_{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\mathbf{g}$ then CSPP is solved by enumerating all paths $\hat{\mathbf{x}}$ such that $\underline{z}(\boldsymbol{\lambda}) \leq (\mathbf{c} + \boldsymbol{\lambda}F)\hat{\mathbf{x}} - \boldsymbol{\lambda}\mathbf{g} \leq \bar{z}$. In turn, this means that, given edge-length vector $\mathbf{c} + \boldsymbol{\lambda}F$, we wish to find all ϵ-optimal (near-shortest) paths for $\epsilon \equiv \bar{z} - \underline{z}(\boldsymbol{\lambda})$. Thus, an NSP algorithm that identifies all ϵ-optimal paths seems more natural for purposes of enumeration than a K-shortest-paths (KSP) algorithm, which is designed to enumerate the K shortest paths for a prespecified integer K. Typical KSP algorithms (e.g., Hadjiconstantinou and Christofides [14]) can and have been used, however, because they enumerate paths in order of increasing length, and can be halted when path length exceeds $(\mathbf{c} + \boldsymbol{\lambda}F)\mathbf{x}^*_{\boldsymbol{\lambda}} + \epsilon$. But, enumerating paths in order of length requires unnecessary computational work and algorithmic complexity; the NSP algorithm of [7] is much simpler.

3 A New LRE Algorithm for CSPP

We can now describe our basic LRE approach for solving CSPP:

LRE Algorithm for CSPP, Outline

1. Optimise the Lagrangian lower bound to find a "good" vector $\boldsymbol{\lambda}$.

2. In the optimisation process, identify a feasible path $\hat{\mathbf{x}}$, and hence an upper bound on z^*, i.e., $\bar{z} \equiv \mathbf{c}\hat{\mathbf{x}} \geq z^*$.

3. Using a standard path-enumeration algorithm, begin enumerating all paths $\hat{\mathbf{x}}$ such that $(\mathbf{c} + \boldsymbol{\lambda}F)\hat{\mathbf{x}} - \boldsymbol{\lambda}\mathbf{g} \leq \bar{z}$, with the following modifications:

(a) Use the side constraints to limit the enumeration when it can be projected that any one of them cannot be satisfied; that is, do not extend the current s-u subpath along an edge (u, v) if that must lead to the violation of one or more side constraints. Similarly, do not extend the subpath along (u, v) if the length of that path, with respect to true edge lengths $\mathbf{c}$, cannot be shorter than $\bar{z}$. (This is handled by creating an extra side constraint $\mathbf{cx} \leq \bar{z}$.)

(b) Update the incumbent solution $\hat{\mathbf{x}}$ and the upper bound $\bar{z} = \mathbf{c}\hat{\mathbf{x}}$ whenever a shorter, feasible path is found.

4. The best solution, $\hat{\mathbf{x}}$, identified in this way, is optimal.

As with many optimization procedures, we can easily modify this LRE algorithm to identify a near-optimal solution, i.e., a solution within $\delta > 0$ units of optimality rather than an exact solution; and this may reduce the computational burden substantially. The pseudo-code for the algorithm listed below does include such a parameter δ, but we do not explore its usefulness in this paper.

The NSP algorithm upon which we base this procedure first computes the minimum distance $d'(v)$ from each $v \in V$ back to t by solving a single, backwards, shortest-path problem starting from t (with respect to unmodified edge lengths c_{uv} in this description). Then, a standard path-enumeration algorithm is begun from s, but the current s-u subpath is extended along an edge (u, v) if and only if (i) the length of the current subpath, denoted $L(u)$, plus c_{uv}, plus $d'(v)$, does not exceed the definition of "near-shortest", and (ii) the path does not loop back on itself. This particular implementation may require exponential work per path enumerated because, after adding an edge to the current subpath, it does not recompute distances $d'(w)$, for $w \in V$. A theoretically efficient implementation must recompute these values which are defined through paths that do not intersect the current subpath and thus can change as the algorithm proceeds. (However, the values $d'(w)$ used in the implemented algorithm represent lower bounds on the "true values", and lower bounds are sufficient to ensure correctness.) This theoretically inefficient implementation turns out to be the most efficient in practice, however [7].

We now provide a detailed description of our LRE algorithm assuming that $\underline{z}(\boldsymbol{\lambda})$ has already been optimised and an incumbent solution $\hat{\mathbf{x}}$ identified.

Path-Enumeration Subroutine for LRE Algorithm to Solve CSPP

DESCRIPTION: Finds a path, identified by its edge-incidence vector $\mathbf{x}^*$, that satisfies $F\mathbf{x}^* \leq \mathbf{g}$ and is within δ units of being shortest.

INPUT: A directed graph $G = (V, E)$ in adjacency-list format, s, t, edge length vector $\mathbf{c} \geq \mathbf{0}$, side-constraint data for $F\mathbf{x} \leq \mathbf{g}$ with F non-negative, optimal or near-optimal Lagrangian vector $\boldsymbol{\lambda}$ for **CSPLR**, incumbent solution $\hat{\mathbf{x}}$, lower bound $\underline{z}(\boldsymbol{\lambda})$ and parameter $\delta \geq 0$.

OUTPUT: A δ-optimal shortest path $\mathbf{x}^*$ satisfying $F\mathbf{x}^* \leq \mathbf{g}$.

NOTE: "firstEdge(v)" points to the first edge in a linked list of edges directed out of v.

{

$\mathbf{x}^* \leftarrow \hat{\mathbf{x}}$; $\bar{z} \leftarrow \mathbf{c}\mathbf{x}^*$; /* $\mathbf{x}^*$ stores the incumbent solution rather than $\hat{\mathbf{x}}$ */

$\mathbf{c}' \leftarrow \mathbf{c} + \boldsymbol{\lambda} F$;
/* Add a "0-th side constraint" to limit enumeration based on $\mathbf{c}$ */
$I^+ \leftarrow I \cup \{0\}$; $\mathbf{f}_0 \leftarrow \mathbf{c}$; $g_0 \leftarrow \bar{z}$;
/* The following requires just one backwards shortest-path calculation */
for (all $v \in V$) { $d'(v) \leftarrow$ minimum distance, in terms of $\mathbf{c}'$, from v to t; }
for (each side constraint $i \in I^+$){
    /* Solve a backwards shortest-path problem using edge "lengths" $\mathbf{f}_i$ */
    for (all $v \in V$) { $d'_i(v) \leftarrow$ minimum weight, in terms of $\mathbf{f}_i$, from v to t; }
}
for(all $v \in V$) { nextEdge(v) $\leftarrow$ firstEdge(v); }
$L(s) \leftarrow -\boldsymbol{\lambda}\mathbf{g}$; /* Initialize path length with the Lagrangian constant term */
for(all $i \in I^+$) { $L_i(s) \leftarrow 0$; } /* Initial path weight with respect $\mathbf{f}_i$ is 0 */
theStack $\leftarrow s$; onStack(s) $\leftarrow$ true; onStack(v) $\leftarrow$ false $\forall\ v \in V \backslash \{s\}$;
while (theStack is not empty){
    $u \leftarrow$ vertex at the top of theStack;
    if (nextEdge(u) $\neq \emptyset$) {
        $(u, v) \leftarrow$ the edge pointed to by nextEdge(u);
        increment nextEdge(u);
        if ((onStack(v) = false)
            and ($L(u) + c'_{uv} + d'(v) < \bar{z} - \delta$)
            and ($L_i(u) + f_{iuv} + d'_i(v) \leq g_i\ \forall\ i \in I^+$)) {
            if ($v = t$) { /* An improved solution has been found */
                Represent the feasible path encoded as theStack $\cup \{t\}$ through
                its edge-incidence vector $\hat{\mathbf{x}}$;
                $\bar{z} \leftarrow \mathbf{c}\hat{\mathbf{x}}$; $g_0 \leftarrow \bar{z}$; $\mathbf{x}^* \leftarrow \hat{\mathbf{x}}$;
                /* Preemptive termination is possible in the following step */
                if ($\bar{z} - \underline{z}(\boldsymbol{\lambda}) \leq \delta$) **goto** Finish;
            } **else** {
                push v on theStack; onStack(v) $\leftarrow$ true;
                $L(v) \leftarrow L(u) + c_{uv}$;
                for (all side constraints $i \in I^+$) { $L_i(v) \leftarrow L_i(u) + f_{iuv}$; }
            }
        }
    } **else** {
        Pop u from theStack; onStack(u) $\leftarrow$ false;
        nextEdge(u) $\leftarrow$ firstEdge(u);
    }
}
Finish: Print ($\mathbf{x}^*$);
}

4 Computational Results

All computational tests are carried out on a laptop computer with an Intel 1 GHz Pentium III processor, 512 megabytes of RAM, the Microsoft Windows 2000 operating system, and with programs written and compiled with the Microsoft Visual

C++ Version 6.0. We compare our results using some of the grid networks studied by Dumitrescu and Boland [10] whose computer has 512 megabytes of RAM and a Pentium III processor running at 930 MHz. Thus, direct comparisons of run times are reasonably fair (or the reader may multiply the times from [10] by roughly 0.9).

The test networks, denoted "Grid(a, b)", are formed on a rectangular grid, a vertices tall and b vertices wide, with a separate source vertex s and sink vertex t external to the grid. The source s is connected to each vertex in the leftmost column of the grid, and each vertex in the rightmost column is connected to t. Each vertex u within the grid has (up to) three edges (u, v) directed out of it, up, down, and from left to right, as long as the vertex v exists in the grid. Edge lengths and weights are uniform, randomly generated integers in the range [1,10] for vertical edges, and in the range [80,100] for horizontal edges. The weight limit is $\lfloor 0.5(g_{\max} + g_{\min}) \rfloor$ where $g_{\min}$ is the weight of the minimum-weight path, and $g_{\max}$ is the weight of the shortest s-t path. This problem structure represents that of "problem class 4-M, Type 2", which is the most computationally difficult class of grid networks generated and studied by Dumitrescu and Boland [10]. (These authors do analyze digital elevation models, which are grid networks taken from other sources, i.e., not generated by the authors.) Table 1 displays computational results for five instances of this problem class of the sizes reported in [10], and one larger one.

Problem	$\|V\|$	$\|E\|$	Run Time (sec)	D&B Run Time (sec)
Grid$(30, 100)$	3,002	8,830	0.04	0.04
Grid$(100, 100)$	10,002	29,990	0.19	0.09
Grid$(200, 200)$	40,002	119,800	0.42	1.53
Grid$(350, 200)$	70,002	209,950	0.70	5.67
Grid$(450, 300)$	135,002	404,850	1.43	20.93
Grid$(1000, 1350)$	1,350,002	4,049,350	14.89	–

Table 1: Computational results for difficult grid networks. Graph structure and numerical data correspond to problem class 4-M, Type 2, in [10]. Solutions are guaranteed to be within 0.1% of optimal, i.e., the "relative optimality tolerance" is 0.1%. Computations are performed on a 1 GHz Pentium III computer. "D&B Run Time", shown for comparison, is the best run time listed in Dumitrescu and Boland [10]. Their computations are performed on a 930 MHz Pentium III computer using a modified label-setting algorithm combined with preprocessing; their relative optimality tolerance is 5%. Those authors do not test a problem of size comparable to Grid$(1000, 1350)$.

It appears from Table 1 that the LRE approach can deliver an order of magnitude speedup over the vertex-labeling algorithm developed in [10], at least for larger problems. This computational testing is preliminary, of course, and a wider variety of problems must be investigated.

5 Conclusions

We have described an improved algorithm for solving the constrained shortest-path problem (CSPP). Our "LRE algorithm" Lagrangianises all side constraints, optimises the resulting Lagrangian function, defines new edge lengths through the Lagrangian function, and enumerates all near-shortest paths in order to close any remaining duality gap. Testing on singly constrained problems indicates that the LRE methodology can be an order of magnitude faster than the most recently published vertex-labeling algorithm.

The LRE algorithm certainly warrants further testing and refinement. Although the problems tested here represent computationally difficult grid networks from [10], these problems do not have large duality gaps (less than 1%), and are therefore not very difficult for our algorithm. A subsequent paper will investigate problems with larger duality gaps. We will also perform testing on the digital elevation models investigated by Dumitrescu and Boland [10], which form the only other class of problems those authors find computationally challenging. Problems with multiple side constraints will also be investigated.

Future refinements of our LRE algorithm will incorporate preprocessing to eliminate parts of the network that cannot appear in any optimal solution. For instance, by solving a shortest-path problem from s with respect to edge weights, and by solving another one backward from t, one may eliminate vertices and edges whose traversal would ensure the violation of at least one side constraint. Our algorithm will not gain as much from such preprocessing as does a vertex-labeling algorithm because the enumeration subroutine automatically avoids searching parts of the network that lead to constraint violations (or whose traversal ensures that any resulting solution must be worse than the incumbent solution). Nonetheless, useful improvements may be gained for some problems.

We are also pursuing applications of the LRE algorithm to column-generation problems that arise in set-partitioning models for airline crew scheduling. Furthermore, we believe that the algorithm holds promise for automated, minimum-risk routing of military aircraft and will be pursuing that application.

The paradigm of Lagrangian relaxation and enumeration is clearly valid for other optimisation problems, including the "max-flow network interdiction problem" (Wood [24]). This problem seeks to reduce, as much as possible, the maximum s-t flow in a capacitated network by interdicting (destroying) a set of edges using a limited interdiction budget. Derbes [8] uses Lagrangian relaxation to solve this problem approximately by identifying a minimum-capacity cut in the network whose capacities are modified through the Lagrangian relaxation. Balcioglu and Wood [3] describe a fast algorithm for enumerating near-minimum-capacity cuts, and that can be used to close any duality gap from the Lagrangian procedure, just as the near-shortest-path enumeration algorithm closes the duality gap in CSPP. We have already begun to investigate this procedure.

Acknowledgments

The authors thank the Air Force Office of Scientific Research, the Office of Naval Research and the Naval Postgraduate School sabbatical program for funding this research. Kevin Wood also thanks the University of Auckland, Department of Engineering Science, for providing support in the preparation of this paper.

References

[1] Ahuja, R.K., Magnanti, T.L., and J.B. Orlin, (1993), *Network Flows*, Prentice Hall, Englewood Cliffs, New Jersey.

[2] Aneja, Y., Aggarwal, V., and Nair, K., (1983), "Shortest Chain Subject to Side Conditions", *Networks*, 13, pp. 295–302.

[3] Balcioglu, A. and Wood, K., (2003), "Enumerating Near-Min s-t Cuts", in *Network Interdiction and Stochastic Integer Programming*, D.L. Woodruff, editor, Kluwer Academic Publishers, Boston, pp. 21–49.

[4] Beasley, J. and Christofides, N., (1989), "An Algorithm for the Resource Constrained Shortest Path Problem", *Networks*, 19, pp. 379–394.

[5] Benders, J.F., (1962), "Partioning Procedures for Solving Mixed Integer Variables Programming Problems", *Numerische Mathematik*, 4, pp. 238–252.

[6] Boerman, D., (1994), "Finding an Optimal Path Through a Mapped Minefield", Master's Thesis, Naval Postgraduate School, Monterey, California, March.

[7] Carlyle, W.M. and Wood, R.K., (2003), "Near-Shortest and K-Shortest Simple Paths", in review; available at http://www.nps.navy.mil/orfacpag/resumePages/Wood-pubs/CarlyleWoodPaths.pdf

[8] Derbes, D., (1997), "Efficiently Interdicting A Time-Expanded Transshipment Network", Master's Thesis, Naval Postgraduate School, Monterey, California, September.

[9] DeWolfe, D., Stevens, J. and Wood, K., (1993), "Setting Military Reenlistment Bonuses", *Naval Research Logistics*, 40, pp. 143–160.

[10] Dumitrescu, I. and Boland, N., (2003), "Improved Preprocessing, Labeling and Scaling Algorithm for the Weight-Constrained Shortest Path Problem", *Networks*, 42, pp. 135–153.

[11] Fox, B.L. and Landi, D.M., (1970), "Searching for the Multiplier in One-Constraint Optimization Problems", *Operations Research*, 18, pp. 253-262.

[12] Gamache, M., Soumis, F., Marquis, M. and Desrosiers, J., (1999), "A Column Generation Approach for Large-Scale Aircrew Rostering Problems", *Operations Research*, 47, pp. 247–263.

[13] Garey, M.R. and Johnson, D.S., (1979), *Computers and Intractability: A Guide to the Theory of of NP-Completeness*, W.H. Freeman and Co., San Francisco.

[14] Hadjiconstantinou, E. and Christofides, N., (1999), "An Efficient Implementation of an Algorithm for Finding K Shortest Simple Paths", *Networks*, 34, pp. 88–101.

[15] Handler, G. and Zang, I., (1980), "A Dual Algorithm for the Constrained Shortest Path Problem", *Networks*, 10, pp. 293–310.

[16] Joksch, H., (1966), "The Shortest Route Problem with Constraints", *Journal of Mathematical Analysis and Application,* 14, pp. 191–197.

[17] Kaufman, D.E. and Smith, R.L., (1993), "Fastest Paths in Time-Dependent Networks for Intelligent Vehicle-Highway Applications", *IVHS Journal*, 1, pp. 1–11.

[18] Korkmaz, T. and Krunz, M., (2001), "Multi-Constrained Optimal Path Selection", in *Proceedings of the IEEE INFOCOM 2001 Conference, Vol. 2*, Anchorage, Alaska, April 22-26, pp. 834–843.

[19] Latourell, J.L., Wallet, B.C. and Copeland, B., (1998), "Genetic Algorithm to Solve Constrained Routing Problems with Applications for Cruise Missile Routing", in *Applications and Science of Computational Intelligence*, *Proceedings of SPIE, Vol. 3390*, Society of Photo-optical Instrumentation Engineers, Bellingham, Washington, USA, pp. 490–500.

[20] Lee, S.H.K., (1995), "Route Optimization Model for Strike Aircraft", Master's Thesis, Naval Postgraduate School, Monterey, California, September.

[21] Nachtigall, K., (1995), "Time Depending Shortest-path Problems with Applications To Railway Networks", *European Journal of Operational Research*, 83, pp. 154–166.

[22] Nygaard, R., Melnikov, G. and Katsaggelos, A.K., (2001), "A Rate Distortion Optimal ECG Coding Algorithm", *IEEE Transactions on Biomedial Engineering,* 48, pp. 28–40.

[23] Vance, P.H., Barnhart, C., Johnson, E.L. and Nemhauser, G.L., (1997), "Airline Crew Scheduling: A New Formulation and Decomposition Algorithm", *Operations Research*, 45, pp. 188–200.

[24] Wood, K., (1993), "Deterministic Network Interdiction", *Mathematical and Computer Modeling*, 17, pp. 1–18.

[25] Zabarankin, M., Uryasev, S. and Pardalos, P., (2001), in *Cooperative Control and Optimization*, R. Murphey and P. Pardalos, eds., Kluwer Academic Publishers, Dordrecht, pp. 273–299.